Real-time Watercolor Painting on a Distributed Paper Model

Tom Van Laerhoven Jori Liesenborgs
Frank Van Reeth
Limburgs Universitair Centrum
Expertise Centre Digital Media
Universitaire Campus
B-3590 Diepenbeek, Belgium
tom.vanlaerhoven@luc.ac.be, jori@lumumba.luc.ac.be, frank.vanreeth@luc.ac.be

Abstract

Current watercolor painting systems produce convincing results, but trade this quality for a simulation that is much too slow for interactive use. We describe the implementation of a paper model that allows for real-time creation of watercolor images by simulating the mechanics of pigment and water throughout a three-layer paper model. In order to keep the simulation real-time while painting on large surfaces, we adopt a distributed paper canvas by dividing it in a grid of subpapers, each of which is delegated to a remote process.

1 Introduction

Digital paint systems provide an artist with the possibility to experiment with various painting media in a more convenient and easier way than traditionally possible. However, the complexity of simulating this process as realistically as possible still causes challenging problems. These problems are apparent in the visual results as well as in the process of creating them.

In this paper we present a distributed paper model that allows users to paint with watercolor in real-time. The model uses a three-layer paper canvas that tries to mimic the real painting process as closely as possible, adopting both physically-based and empirically-based algorithms to simulate paint.

In the next section we start by giving a brief overview of related work on digital paint systems. A description of the paper model starts in section 3. We conclude this paper with some results in section 5 and some closing remarks along with a list of planned future work in section 6.

2 Related Work

Strassman was the first to present a physical model of the brush movement on a paper with the purpose of creating traditional Japanese artwork using black ink [10]. The role of the paper canvas itself, however, which is our main concern in this paper, is limited to rendering the brush's imprint.

This limitation is addressed by Small [7], exploring the actual behavior of pigment and water when applied to paper. The canvas is designed as a two-dimensional grid of interacting cells, also called a cellular automaton [12], a design that was adopted by many authors [15, 14, 1]. The system is implemented on a parallel architecture, dedicating each individual cell to a single processor.

Inspired by [7], [1] also uses a cellular automaton to simulate fluid flow and pigment dispersion. The painting consists of an ordered set of translucent glazes or washes, the results of independent fluid simulations. Due to its complexity, the painting process is not real-time.

The structure of a paper canvas typically consists of a mesh of randomly distributed fibers. In [3], the first software model of such a structure was proposed. Since then a number of improvements were made, taking into account the paper's texture [6] and using more complex diffusion models [5].

Earlier painting methods generate convincing results but trade this quality for a simulation that is much too slow for interactive use. In this paper we combine both properties. Taking the concept of a three-layer paper model, combined with the work on fluid flows from Stam [9] as a starting point, we create a realistic painting simulation, maintaining the real-time property by distributing the paper canvas, dividing computational complexity over several processors. It allows for painting on larger surfaces, and it leaves us with enough computing power to include future extensions.

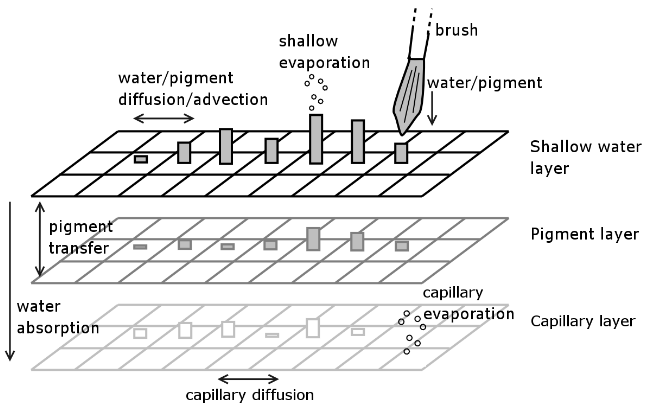

Figure 1. A schematic view of the three-layer paper model with some basic activities.

3 Three-layer Paper Model

In this section we give a brief overview of our paper model. For an in-depth description we refer to [11].

The concept of a paper model consisting of three layers was first introduced by Curtis et al. [1]. This idea makes sense when examining the different states in which water and pigment can exist during painting activity. The paint, a mixture of water and pigment that acts like a fluid flow, is put onto the paper canvas by a brush. The water is then absorbed into the paper and spreads throughout the paper structure, while the pigment particles are deposited on the surface and possibly picked back up by the paint fluid later on. Analyzing this behavior, three different states for pigment or water can be distinguished:

- Pigment and water in a layer on top of the paper.
- Pigment deposited on the paper surface.
- Water absorbed by the paper.

We assign a different layer to each state, each of which is implemented as a two-dimensional grid of cells. Values assigned to a cell can be thought of as sampled in the real paper canvas. Simulating the system through time then consists of running a set of stacked cellular automata. Figure 1 shows a schematic view of the paper model along with some basic activities.

3.1 Shallow Layer

Before paint is absorbed into the paper, it stays on top of the surface where it acts like a shallow layer of water. The velocity at each cell is described by a vector field $(v_x, v_y)_{i,j}$. Scalar fields $w_{i,j}$ and $(p_{\text{idx}})_{i,j}$ contain for each cell the water quantity and the amount of every pigment respectively. A fluid flow in this condition can be simulated by solving the two-dimensional Navier-Stokes equation [4].

A similar shallow layer used in the paper model from Curtis [1] uses the algorithm proposed by Foster and Metaxas [2] to solve this equation. This approach, however, is not suitable for our real-time simulation, as it is both slow and unstable under certain circumstances such as a high viscosity factor, which is what we are dealing with in case of thick paint fluid.

Our method for solving the Navier-Stokes equation to simulate the velocity field mainly follows the fast and stable algorithms described by Stam [9]. The procedure for updating the water and pigment quantities is similar, but because we want to add constraints on the upper and lower boundaries of a cell's water content, and because we want to include a mechanism to simulate the “dark edge” effect, we developed our own algorithms. They are described in [11].

3.2 Pigment Layer

The pigment particles are initially dropped in the shallow layer, but eventually end up being deposited on the surface of the paper canvas. In the meantime, there is a continuous adsorption and desorption of pigment particles between the shallow layer and the pigment layer. This transfer is influenced by the structure of the paper canvas as well as the granulation, density and staining power of each pigment. Details on how we handled this subject can be found in [11].

3.3 Capillary Layer

The capillary layer represents the inner paper structure and is responsible for the movement of absorbed water, allowing a stroke to spread across its original boundaries. At this point in the simulation, water movement is governed by microscopic capillary effects that can be described in terms of diffusion. Our main goal here is the creation of some irregularity in the capillary diffusion process and in the paper surface, which affects the way pigment particles are transferred. Both the surface texture of the paper canvas and the water capacities of each cell are generated using a method for creating procedural textures by [13]. The gray scale texture values are then translated to a height field and a capillary capacity field. In [11], we elaborate on every aspect of the capillary layer.

4 Distributed Model

In order to produce convincing results, the dimensions of the simulation grid must be chosen so that every cell covers at most one pixel. However, the nature of our simulation

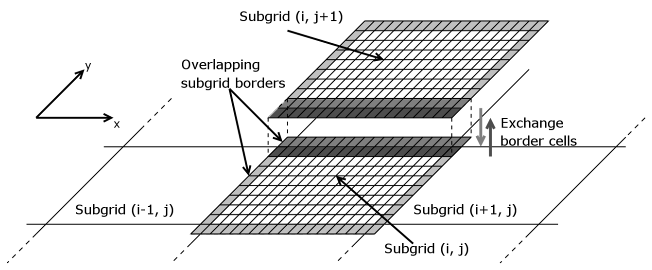

Figure 2. A paper canvas divided in subgrids or subpapers. Each subgrid has an extra border of cells, overlapping neigboring subgrids.

makes it hard to maintain real-time rates if the canvas is too large. For this reason we have chosen to distribute the system by breaking up the whole grid in smaller subgrids that are simulated separately at remote processing units. This approach is valid because the system is constructed as a set of stacked cellular automata, which means that the processing of a single cell only requires knowledge of its four immediate neigbors. All algorithms described in the previous sections, or in more detail in [11], are created to meet these requirements.

We used LAM/MPI [8], an implementation of the Message Passing Interface (MPI), as a software platform for our implementation. It permits a collection of Unix computers hooked together by a network to be used as a single large parallel computer.

4.1 Distributing Data

The grid that represents the whole paper canvas, and actually consists of the three layers we described in earlier sections, is divided as shown in figure 2. Each subgrid needs an extra row of cells at its border, overlapping the neighboring subgrids. This is because the evaluation of the cells in the actual border needs information from the neighboring cells. Each subgrid can be looked upon as a separate piece of paper canvas. Except for the cells at the border it can also be processed separately, so each subgrid or subpaper is sent to a process that runs on a remote computer.

The remote simulation of a subpaper does not differ from that of a local paper. The only difference lies in the treatment of the border cells, whose contents are not calculated by the subpaper itself, but received from a neigboring subpaper. At the beginning of each time step, a subpaper transfers border cells between his immediate neighbors. An example is shown in figure 2, which depicts the transfer between subpaper (i, j) and his top neighbor $(i, j + 1)$.

Not only the simulation occurs remotely, also the rendering step is performed there. The results are sent back to the parent application and combined with the results of other subpapers to produce the final image.

4.2 Distributing Brush Actions

Applying a brush to the paper canvas produces a stroke that consists of a set of interpolated positions. At each of these positions water, pigment and velocity values are assigned to a collection of cells that belong to the brush pattern. These patterns now have to be applied at a remote subpaper. All position data from the stroke is sent to the remote subpapers to which that particular paper position belongs. The subpapers use a copy of the original brush to apply the pattern themselves.

A position belongs to just one subpaper, but when the brush pattern is applied, the cells of several subpapers could be affected. In this situation, the position under consideration is sent to each subpaper that will be affected by the application of the brush pattern.

5 Results

The examples shown in this section were all painted on a grid that maps a cell to a single pixel. The painting process is done in real-time at an initial speed comparable to the frame rate of video.

Figure 6 shows the frame rate while drawing on a 256×256 paper canvas that was simulated locally, in comparison to a canvas that was divided in four and six subpapers. The tests were all performed on Intel(R) Xeon(TM) 2.40GHz computers using scripted brush actions to make the results comparable. We can conclude that the frame rate stays within acceptable boundaries on a distributed canvas.

The examples depicted in figure 3 show the dark edge effect and some overlapping strokes. The paper structure is clearly visible in the darker overlapping area. Figure 4 and figure 5 both show images created with our system. They were painted on a 400×400 sized grid. We used circular brushes with varying sizes, as well as varying amounts of pigment and water. The palette consisted of eight different pigment types.

6 Conclusions and Future Work

We described a real-time simulation for painting with watercolor on a canvas that is distributed over several processing units. The brush model was kept simple but still produced convincing results at real-time, as shown in the results section.

We are currently investigating better input methods, such as haptic feedback devices, as well as the use of programmable graphics hardware to obtain speed-up, removing the current necessity of additional processing units.

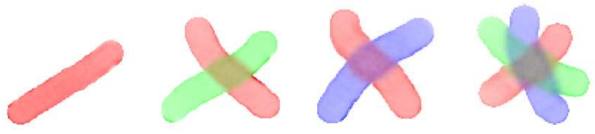

Figure 3. Examples of computer-generated strokes. The dark edge effect in figure 3(a), and overlapping strokes in figures 3(b), 3(c) and 3(d).

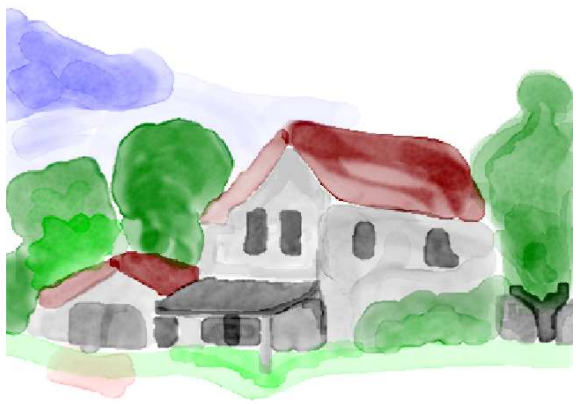

Figure 4. An image with dimension 400×400**.**

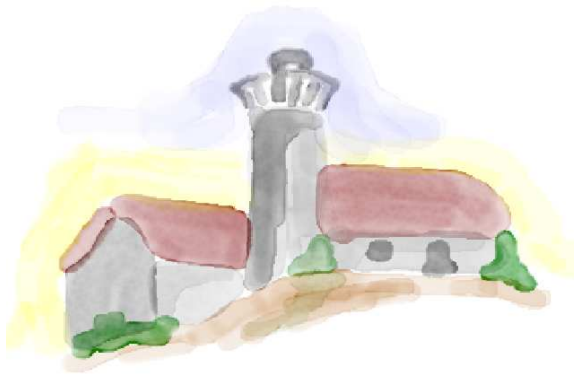

Figure 5. An image with dimension 400×400**.**

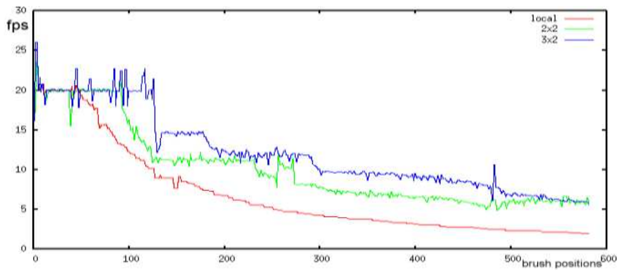

Figure 6. The frame rate while drawing on a local paper canvas (red) and a canvas that was divided in 4 (green) and 6 (blue) subpapers.

References

[1] C. J. Curtis, S. E. Anderson, J. E. Seims, K. W. Fleischer, and D. H. Salesin. Computer-generated watercolor. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 421–430. ACM Press/Addison-Wesley Publishing Co., 1997.

[2] N. Foster and D. Metaxas. Realistic animation of liquids. In *Graphical Models and Image Processing*, pages 471–483, 1996.

[3] Q. Guo and T. L. Kunii. Modeling the diffuse painting of sumie. *IFIP Modeling in Computer Graphics*, 1991.

[4] P. K. Kundu and I. M. Cohen. *Fluid mechanics*. Academic Press, second edition, 2002.

[5] T. L. Kunii, G. V. Nosovskij, and T. Hayashi. A diffusion model for computer animation of diffuse ink painting. In *Proceedings of the Computer Animation '95*, pages 98–102, 1995.

[6] J. Lee. Simulating oriental black-ink painting. volume 19, pages 74–81. IEEE Computer Society Press, 1999.

[7] D. Small. Modeling watercolor by simulating diffusion, pigment, and paper fibers. In *Proceedings of SPIEEE '91*, feb 1991.

[8] J. M. Squyres and A. Lumsdaine. A Component Architecture for LAM/MPI. In *Proceedings, 10th European PVM/MPI Users' Group Meeting*, number 2840 in Lecture Notes in Computer Science, Venice, Italy, sep 2003. Springer-Verlag.

[9] J. Stam. Real-time fluid dynamics for games. In *Proceedings of the Game Developer*, mar 2003.

[10] S. Strassmann. Hairy brushes. In *Proceedings of the 13th annual conference on Computer graphics and interactive techniques*, pages 225–232. ACM Press, 1986.

[11] T. Van Laerhoven, J. Liesenborgs, and F. Van Reeth. A paper model for real-time watercolor simulation. Technical Report TR-LUC-EDM-0403, EDM/LUC, Diepenbeek, Belgium, 2004.

[12] S. Wolfram. *A new kind of science*. Wolfram Media, Inc., first edition, 2002.

[13] S. Worley. A cellular texture basis function. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 291–294. ACM Press, 1996.

[14] Y. J. Yu and D. H. Lee. Interactive rendering technique for realistic oriental painting. In *Journal of WSCG'2003*, volume 11, pages 538–545, 2003.

[15] Q. Zhang, Y. Sato, J. ya Takahashi, K. Muraoka, and N. Chiba. Simple cellular automaton-based simulation of ink behaviour and its application to suibokuga-like 3d rendering of trees. In *Journal of Visualization and Computer Animation*, pages 27–37, 1999.